FIGURE 1

CRYSTAL 5321 — 200A
256K/128K

CRYSTAL 5372 — 200B
512K

MICRO MACHINE
128K
MDATA1

MDATAN

MDI — 110

FILTERING/ DECIMATION ENGINE — 120

SERIAL DATA OUTPUT — 130

FIGURE 2

SELECTABLE VOLTAGE
E.G. 2.5V V_{DD}
DIGITAL SUPPLY PIN

SELECTABLE VOLTAGE E.G. 5.0V I/O PIN	11/25	OPTIONAL LEVEL TRANSLATOR	7/40	OPTIONAL LEVEL TRANSLATOR	5÷	SELECTABLE VOLTAGE E.G. 3.3V I/O POWER PIN VDD PAD 1

MDATA(4:1)

MFLAG(4:1)

DCLK

SDDAT

SDCLK

GROUND PIN FOR 5V I/O AND SUBSTRATE

24/38

SUBSTRATE

3.3V GROUND PIN FOR 3.3V I/O POWER

DIGITAL GROUND PIN

FIGURE 3

FIGURE 4

FIGURE 5

2. Bismarck Sinc Decimation Chain

EXT

SINC 1

**SINC 2**
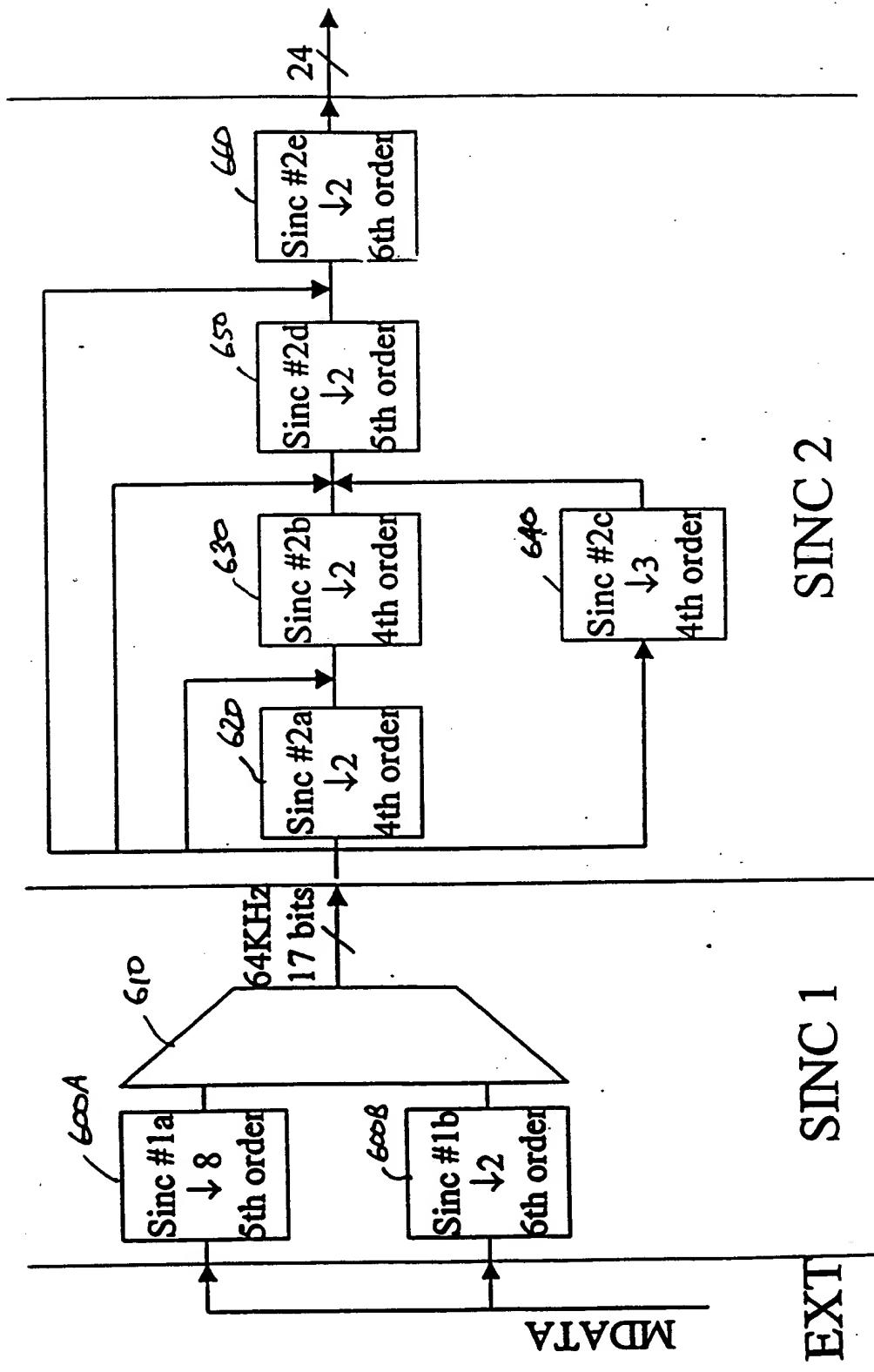
FIGURE 6

- Fifth order decimate by 8:

$$H(z) = \left(\frac{1 - z^{-8}}{1 - z^{-1}} \right)^5$$

- 36 tap FIR filter. Half of the (symmetric) coefficients

| $h_0=1$	$h_1=5$	$h_2=15$	$h_3=35$	$h_4=70$	$h_5=126$	$h_6=210$	$h_7=330$	$h_8=490$
$h_9=690$	$h_{10}=926$	$h_{11}=1190$	$h_{12}=1470$	$h_{13}=1750$	$h_{14}=2010$	$h_{15}=2226$	$h_{16}=2380$	$h_{17}=2460$

FIGURE 7

Figure 8

Channel select

Sinc2 Output Registers

ch1
ch2
ch3
ch4

ROM 1
ROM 2
Head/Tail ROM
accreg

2bits 8bits 8bits 8bits 8bits 8bits 8bits 2bits

HEAD
TAIL 1-4
BYTE 0
BYTE 1
BYTE 2
BYTE 3
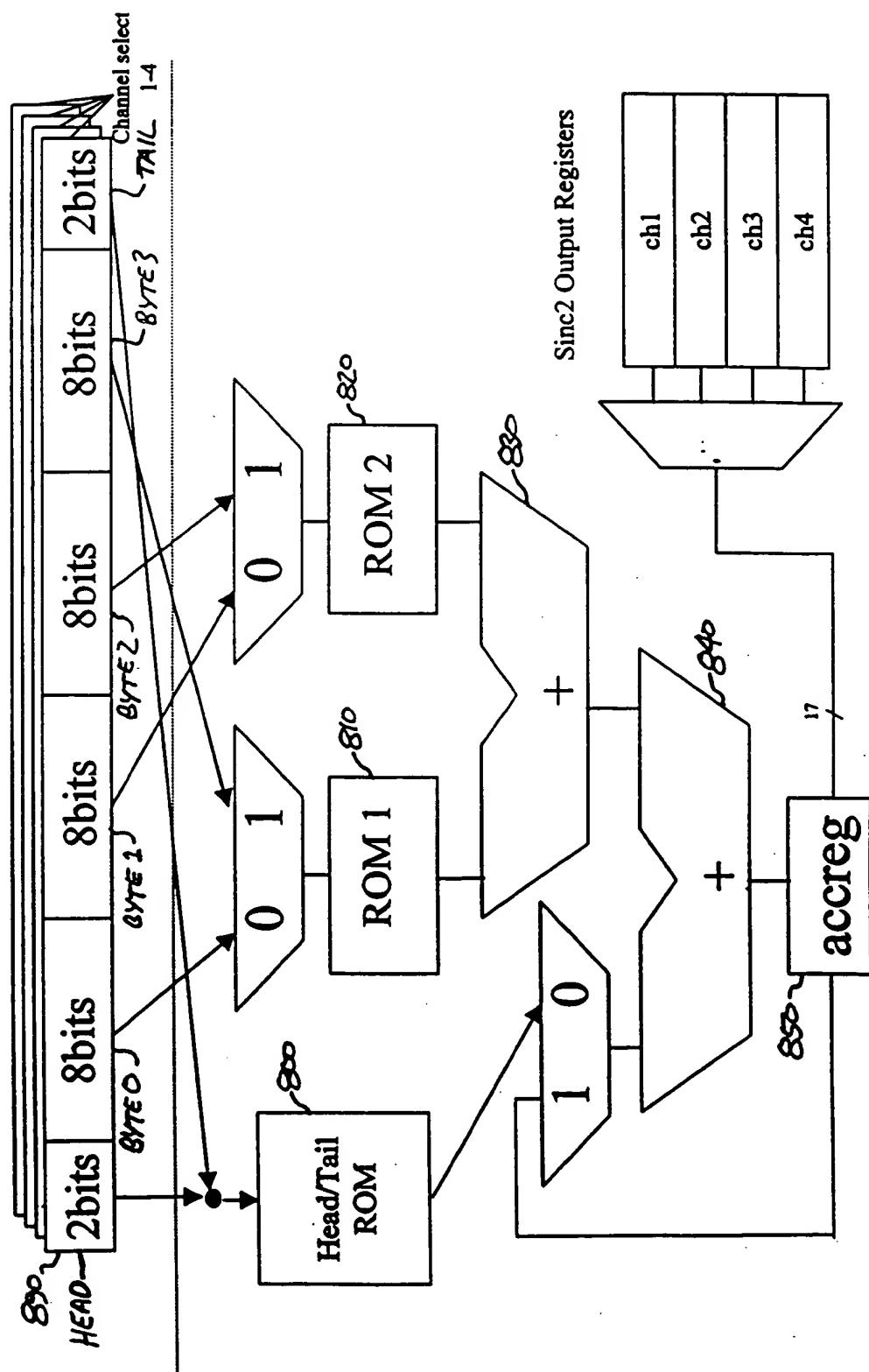
Impulse Response:

$$y[n] = x[n] + 6 \cdot x[n-1] + 15 \cdot x[n-2] + 20 \cdot x[n-3] + 15 \cdot x[n-4] + 6 \cdot x[n-5] + x[n-6]$$

$$H(z) = \left(\frac{1 - z^{-2}}{1 - z^{-1}}\right)^6$$

FIGURE 9

3. Bismarck Sinc1b Functional Diagram

CH1

CH2

CH3

CH4

channelData_mux

1

7 7 7 7

1000

00 01 10 11

3 11 10 01 00

romAddr_accum_mux

1

0

1010

3-Bit
Addressed
ROM

1020

8

Middle
Bit
Converter

1030

romAddr_accum_mux

1

0

1040

8

1050

+

1060

accum_fb

posedge
CLK512

negedge
CLK512

accum_reg

1070

8

channelData_mux

1080

datout1
datout2
datout3
datout4
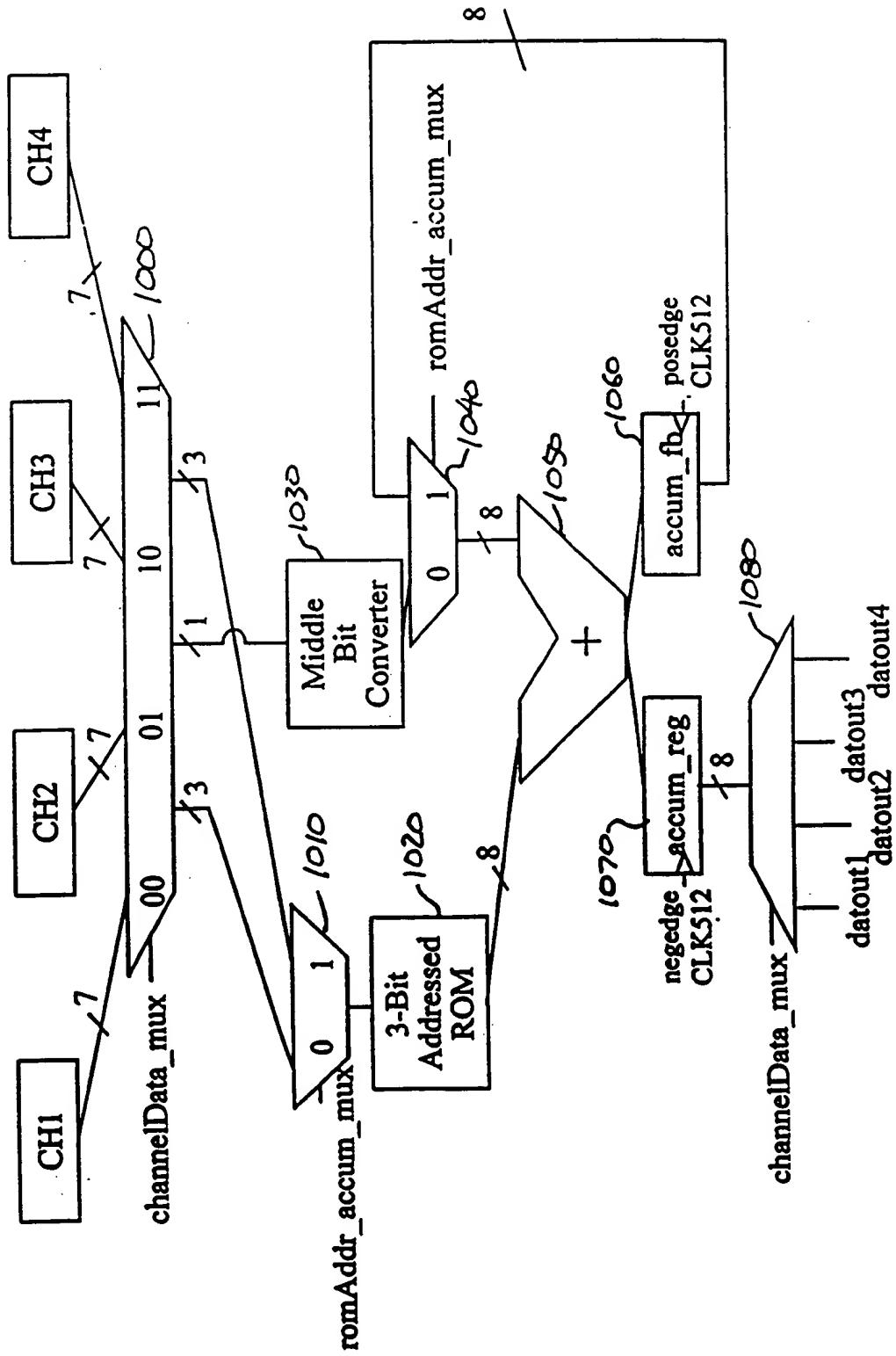
FIGURE 10

Filter Name	System Function	IMPULSE RESPONSE (FILTER COEFFICIENTS)
Sinc2(a) Sinc2(b)	$H(z) = \left(\dfrac{1-z^{-2}}{1-z^{-1}}\right)^4$	$h[n] = [1\ 4\ 6\ 4\ 1]$
Sinc2(c)	$H(z) = \left(\dfrac{1-z^{-3}}{1-z^{-1}}\right)^4$	$h[n] = [1\ 4\ 10\ 16\ 19\ 16\ 10\ 4\ 1]$
Sinc2(d)	$H(z) = \left(\dfrac{1-z^{-2}}{1-z^{-1}}\right)^5$	$h[n] = [1\ 5\ 10\ 10\ 5\ 1]$
Sinc2(e)	$H(z) = \left(\dfrac{1-z^{-2}}{1-z^{-1}}\right)^6$	$h[n] = [1\ 6\ 15\ 20\ 15\ 6\ 1]$

FIGURE 11

Sinc2(a) and Sinc2(b):

$$y[n] = x[n] + 4x[n-1] + 6x[n-2] \qquad + 4x[n-3] + x[n-4]$$
$$= x[n] + 4x[n-1] + 4x[n-2] + 2x[n-2] + 4x[n-3] + x[n-4]$$

x[n]

4x[n-1]

4x[n-2]

2x[n-2]

4x[n-3]

x[n-4]

+

| y[n] |
|------|
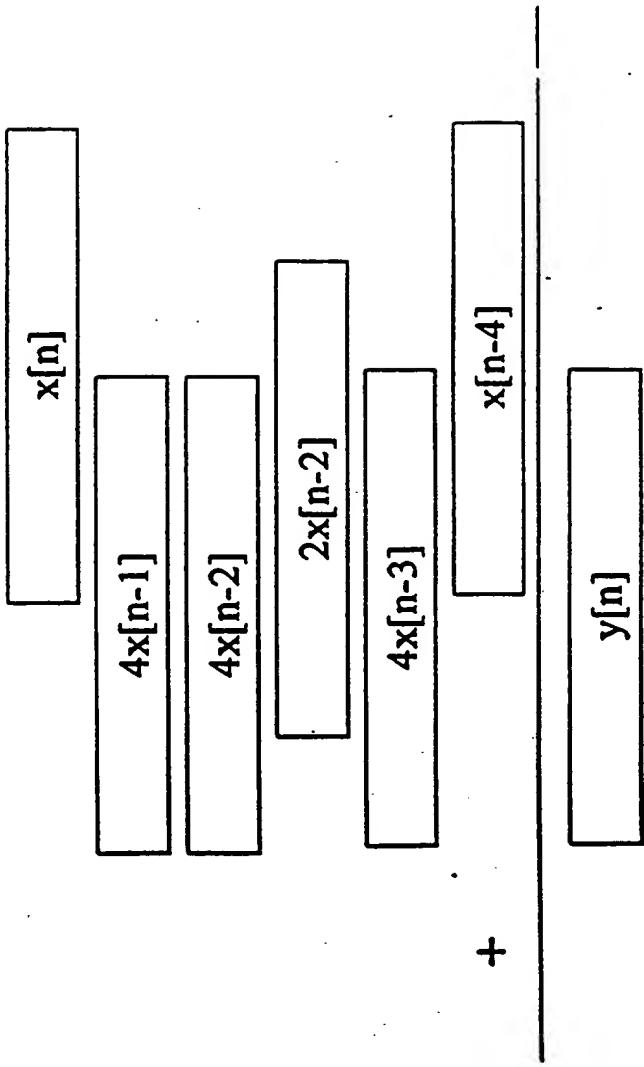
FIGURE 12

Figure 13A

Sinc2(c):

$$y[n] = x[n] + 4x[n-1] + 10x[n-2] + 16x[n-3] + 19x[n-4] + 16x[n-5] + 10x[n-6] + 4x[n-7] + x[n-8]$$

$$= x[n] + 4x[n-1] + \boxed{8x[n-2] + 2x[n-2]} + \boxed{16x[n-3] + 16x[n-4] + 2x[n-4] + x[n-4]}$$
$$+ 16x[n-5] + \boxed{8x[n-6] + 2x[n-6]} + 4x[n-7] + x[n-8]$$

Figure 13B

Sinc2(d):

$$y[n] = x[n] + 5x[n-1] + 10x[n-2] + 10x[n-3] + 5x[n-4] + x[n-5]$$

$$= x[n] + \boxed{4x[n-1] + x[n-1]} + \boxed{8x[n-2] + 2x[n-2]} + \boxed{8x[n-3] + 2x[n-3]} + \boxed{4x[n-4] + x[n-4]} + x[n-5]$$

Figure 13C

Sinc2(e):

$$y[n] = x[n] + 6x[n-1] + 15x[n-2] + 20x[n-3] + 15x[n-4] + 6x[n-5] + x[n-6]$$

$$= x[n] + \boxed{4x[n-1] + 2x[n-1]} + \boxed{16x[n-2] - x[n-2]} + \boxed{16x[n-3] + 4x[n-3]} + \boxed{16x[n-4] - x[n-4]} + \boxed{4x[n-5] + 2x[n-5]} + x[n-6]$$

Sinc2(a) and Sinc2(b):
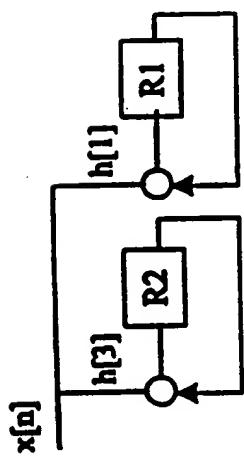

Accumulate Phase (2 additions)

FIGURE 14A
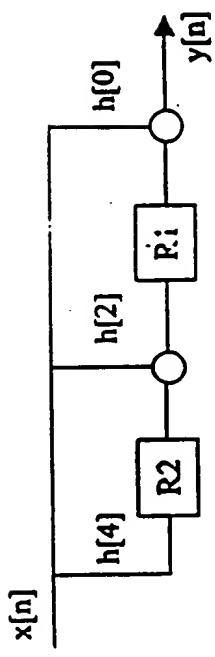
Output Phase (4 additions)

FIGURE 14B

Sinc2(d):
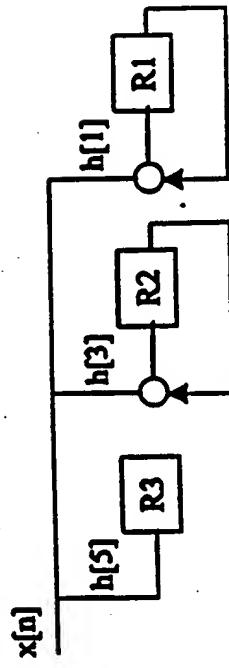
Accumulate Phase (5 additions)

FIGURE 15A
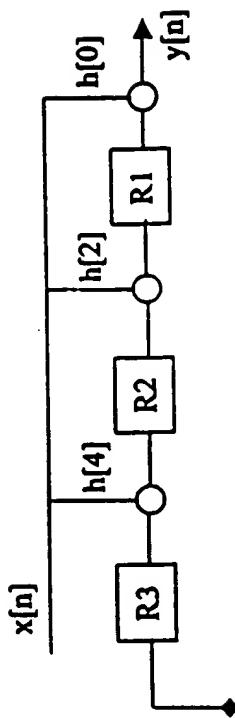
Output Phase (5 additions)

FIGURE 15B

Sinc2(e):
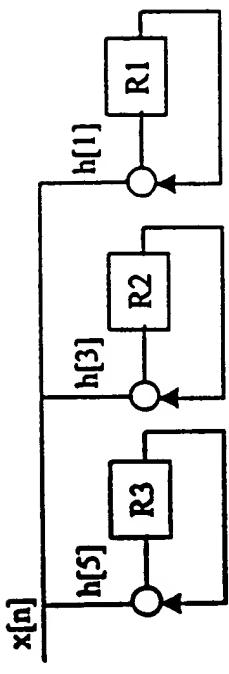
Accumulate Phase (6 additions)

FIGURE 17A
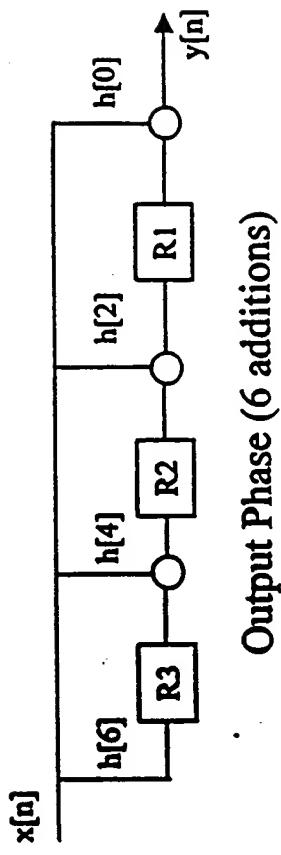
Output Phase (6 additions)

FIGURE 17B
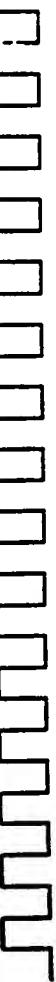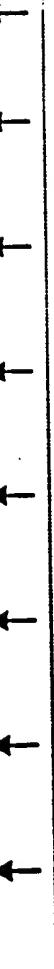
CLK64

2aA(2)

2aO(4)

2bA(2)

2bO(4)

2dA(5)

2dO(5)

2eA(6)

2eO(6)

FIGURE 18A

FIGURE 18B

FIGURE 18C

FIGURE 18D

FIGURE 18E

FIGURE 18F

FIGURE 18G

FIGURE 18H

FIGURE 18I

RAM2
Subroutines

| 2 Accumulate (2 lines) |
| 2 Output (4 lines) |
| .... |
| 5 Accumulate (6 lines) |
| 5 Output (6 lines) |

1910

FIGURE 19B

RAM1
Main Program

| 2aA(2) 2dO(5) |
| 2aO(4) 2bA(2) |
| .... |
| 2aA(2) |
| 2aO(4) 2bO(4) |

1900
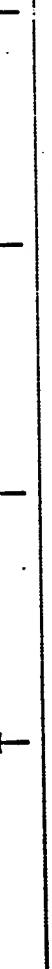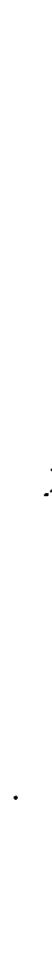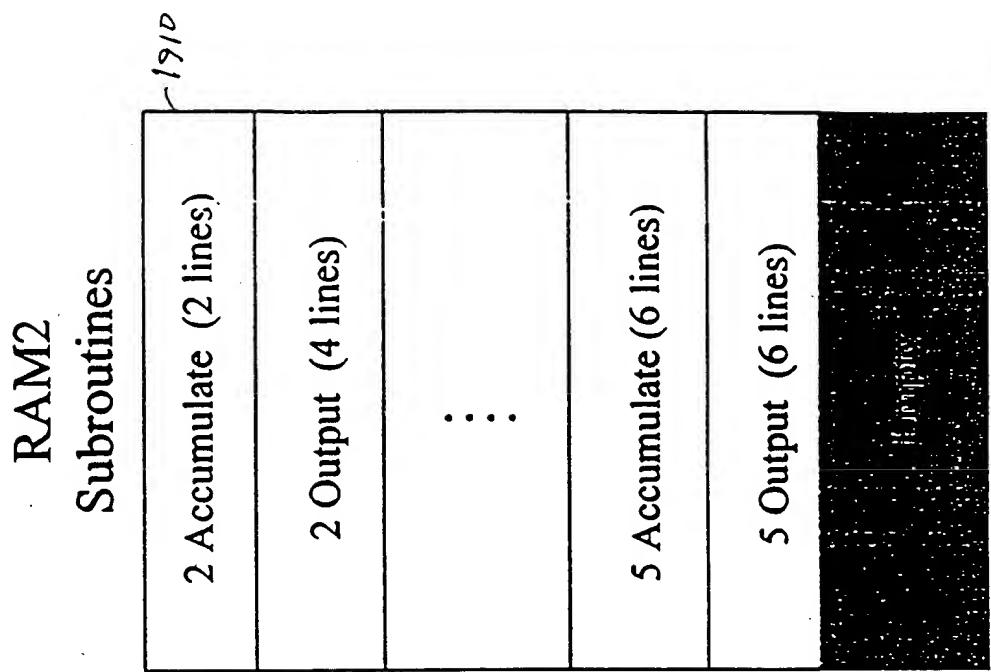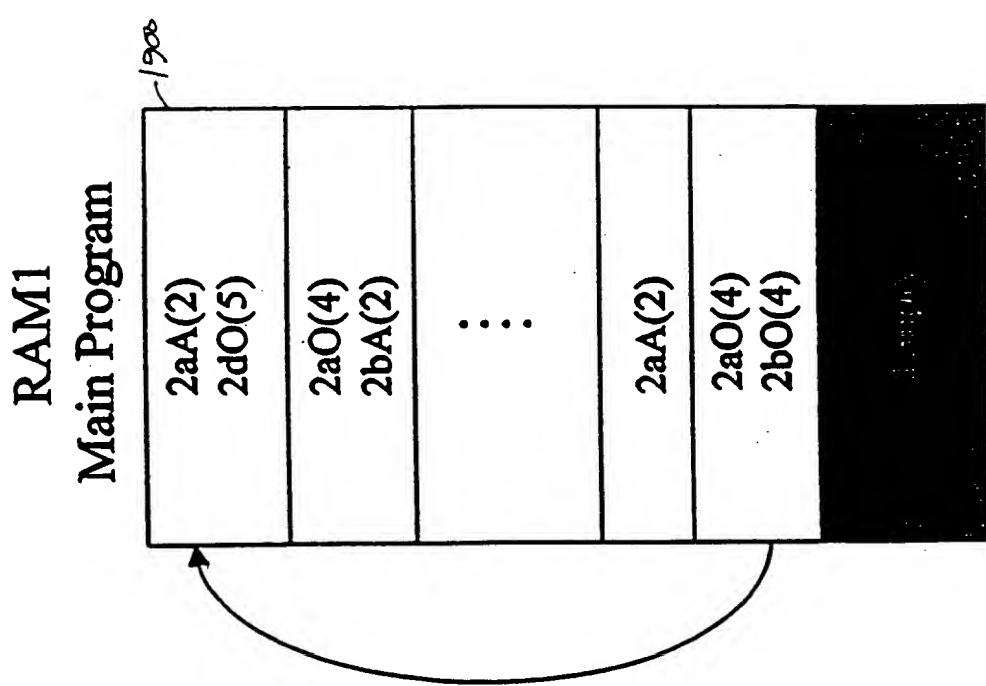
FIGURE 19A

Sinc2 Control-Datapath Architecture

Data from Sinc1

Sinc2 Control

FIGURE 20A

DSP DataBus (24-bits)

b0=24'h0
b1
b2
· · ·
b12
b13
b14
b15=dataout

24
dataout

control[5:1]

control[12:9]

control[8:6]

control[13]

control[17:14]

control[5:1]

datin

a0=datin
a1
a2
a3
a4
a5
a6
a7

24

24

24

+ / -
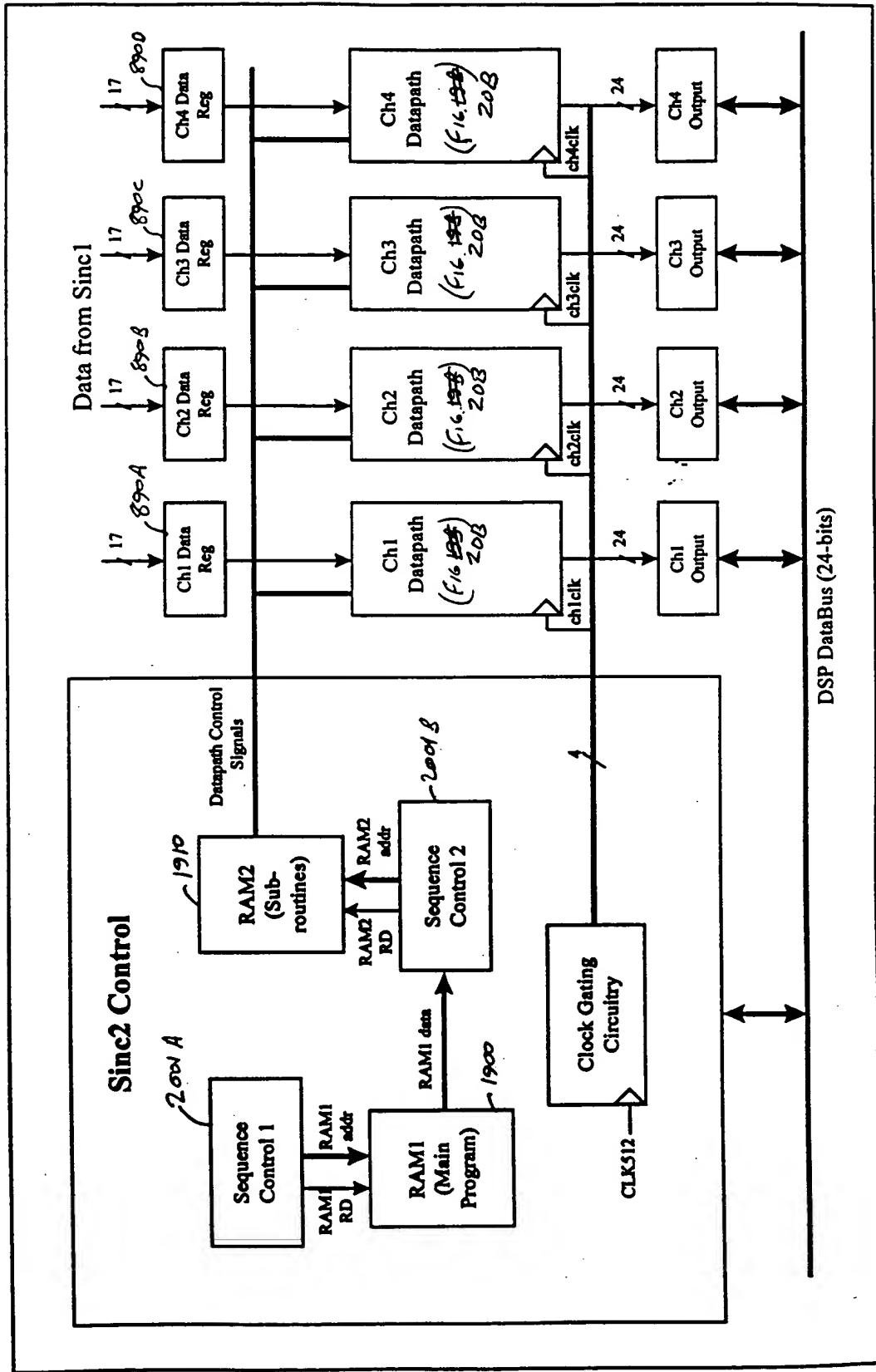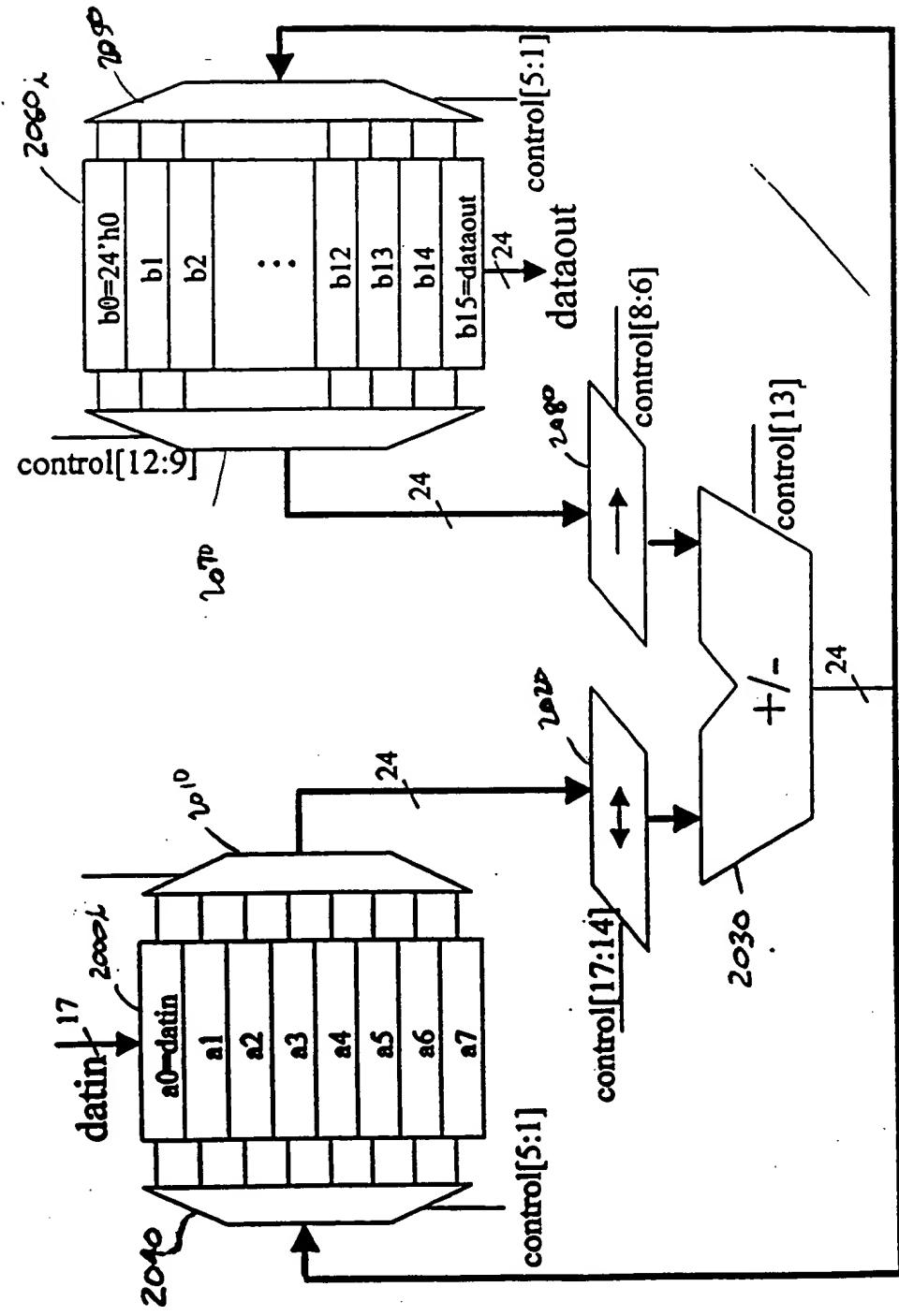
FIGURE 20B

Programming Procedure:

1. Select decimation rate.

2. Select required mini-sincs and associated Accumulate and Output subroutines.

3. Separate coefficients into form suitable for shift-add operations.

4. Check for overflow after each addition in the filter.

5. Perform necessary truncation to 24 bits and scaling of subsequent coefficients in mini-sincs.

6. Time multiplex Accumulate and Output Subroutines so that a maximum of 8 additions/subtractions are performed for each input from sinc1.

7. Create code for RAM2 (Accumulate and Output Subroutines) in the form:
   [Coeff 1] [Src 1] [Src 2] [Dest] [Coeff2] [Done Subroutine]

8. Create code for RAM1 (Main Control code)
   [Line #] [Wait for new data] [Done program]

$Figure \ 21$

From a given 1's density, determine the maximum value entering Sinc2 from Sinc1 (Sinc1a or Sinc1b).
Ex. 93% 1's density fed through Sinc1a provides a maximum value of 0x7FFC fed into Sinc2.

— 2200

Propagate this maximum value through the direct -transpose form of each of the mini-sinc filters making up the filter chain to be programmed in sinc2.
After each adder encountered, check to see if saturation has occurred

~ 2210

2240

In programming this mini-sinc, we prevent overflow by shifting the operands of the saturating addition to the right before addition. Specifically, this implies shifting the result of the previous storage register to the right (sign-extending as needed). We also need to shift all subsequent coefficients in the current mini-sinc to the right by the same number of shifts so that the addition:
    Input*Coefficient + Previous storage register
works properly.

2220

Saturation ?

Yes

No

2230

Done ?

No

Yes

END

2250

FIGURE 22

FIGURE 23

DSP Program Memory Map
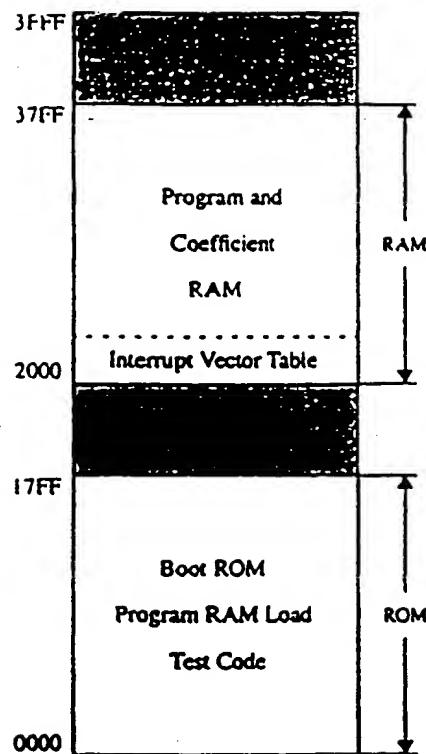
DSP Data Memory/Register Map

```
3FFF  ▓▓▓▓▓▓▓▓▓▓▓▓▓

37FF ─┬───────────────┐  ↑
      │               │
      │  Program and  │
      │               │
      │  Coefficient  │  RAM
      │               │
      │     RAM       │
      │               │
2000 ─┤ Interrupt Vector Table │  ↓
      ▓▓▓▓▓▓▓▓▓▓▓▓▓
17FF ─┼───────────────┤  ↑
      │               │
      │  Boot ROM     │
      │               │
      │ Program RAM Load │  ROM
      │               │
      │   Test Code   │
      │               │
0000 ─┴───────────────┘  ↓
```

```
1FFF ─┬───────────────┐  ↑
      │               │
      │   Data RAM    │  RAM
      │               │
0FFF ─┼───────────────┤  ↓
      │   Data ROM    │  ↑
      │   Sine Table  │  ROM
      │ Pulse/Test Oscillator │
0000 ─┴───────────────┘  ↓
```
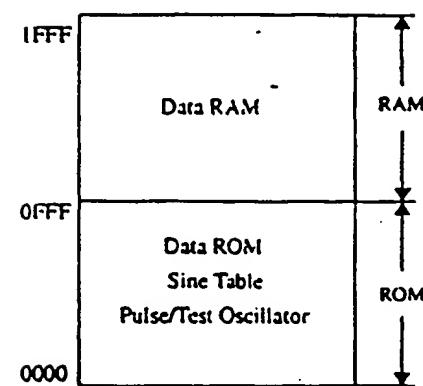
FIGURE 24A
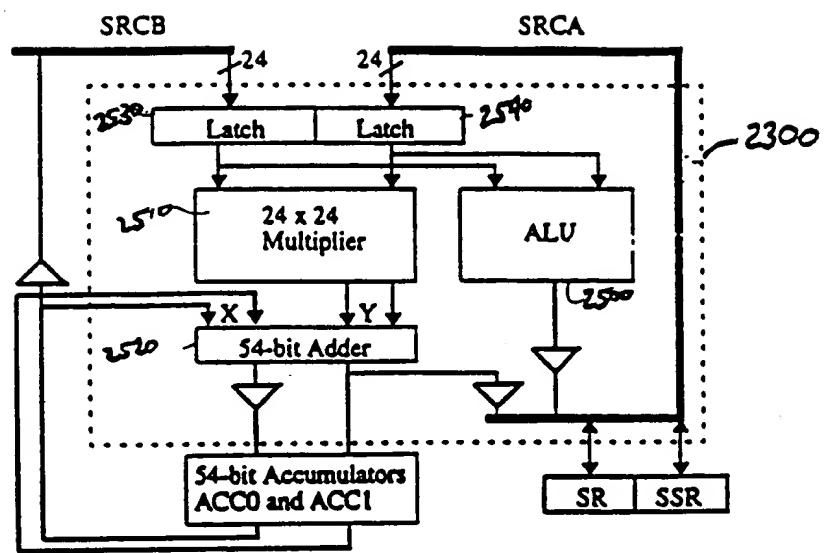
FIGURE 24B

FIGURE 25

FIGURE 26

FIGURE 27

FADDR[1]	FADDR[0]	CH[1]	CH[0]	–	TB	V	W
31	30	29	28	27	26	25	24

Output Word

SDDAT ← | STATUS | DATA |
31   25 24 23   0

23   17   0
SD_STAT (0x17)

W

23   0
SD_DATA (0x18)

FIGURE 28

Microcontroller
Token Out — SDTKI / 100
Ready — SDRDY
Data In — SDDAT
Clock Out — SDCLK
Token In — SDTKO
2900

SDTKI / 100
SDRDY
SDDAT
SDCLK
SDTKO

SDTKI / 100
SDRDY
SDDAT
SDCLK
SDTKO
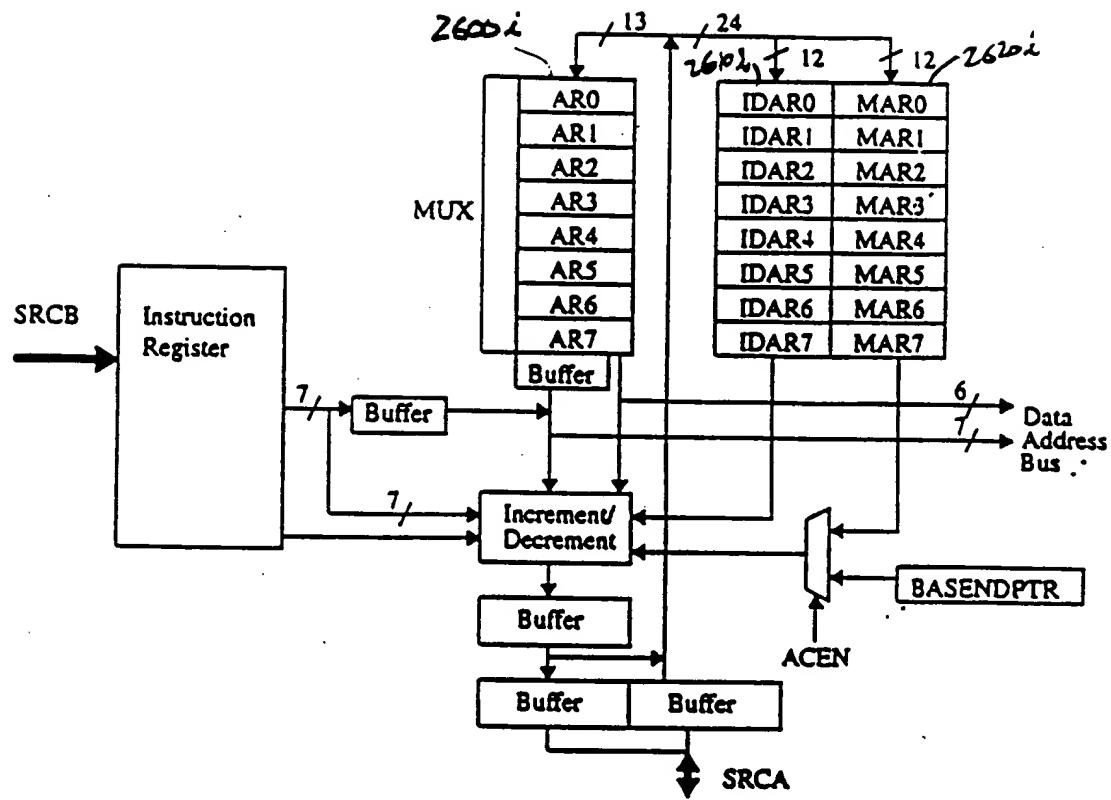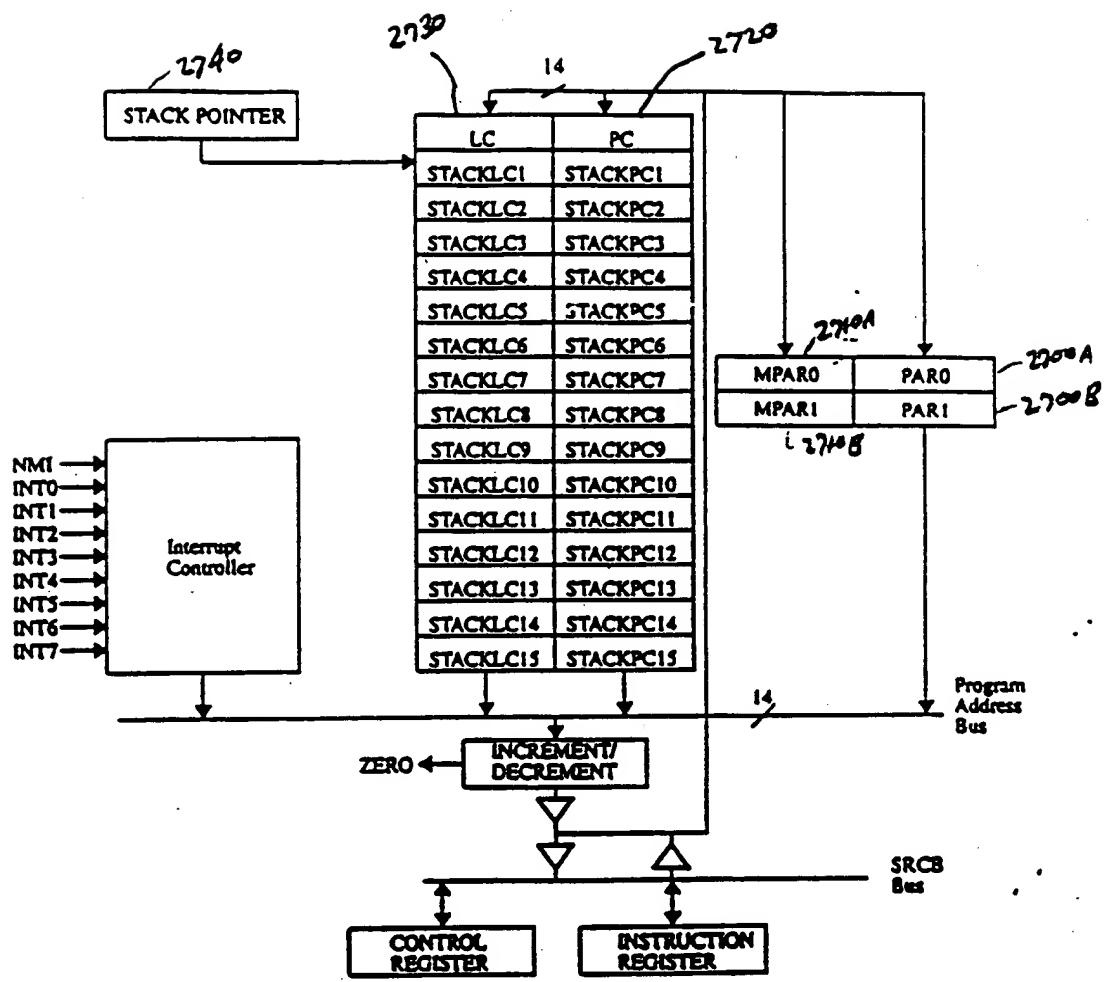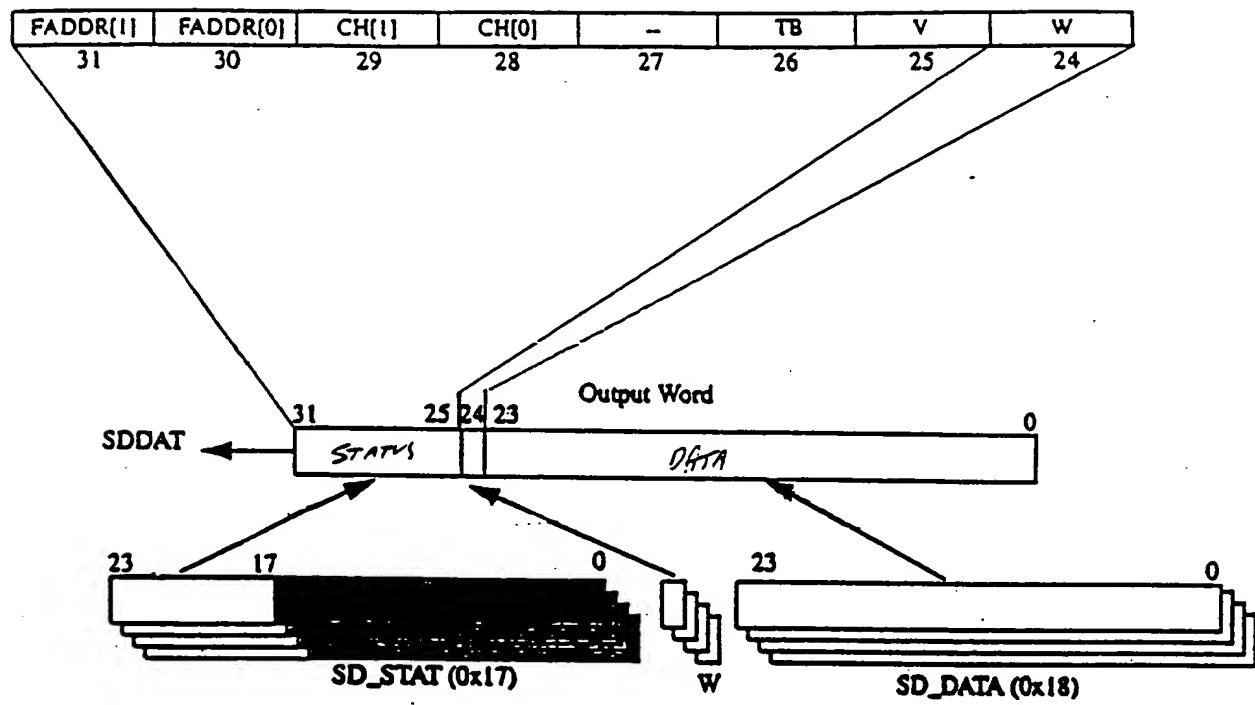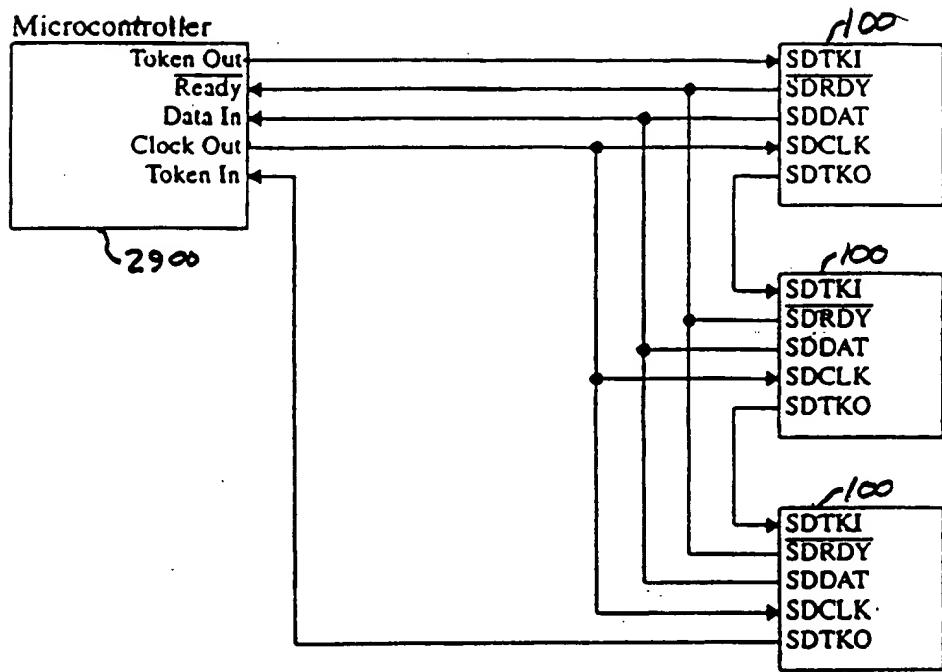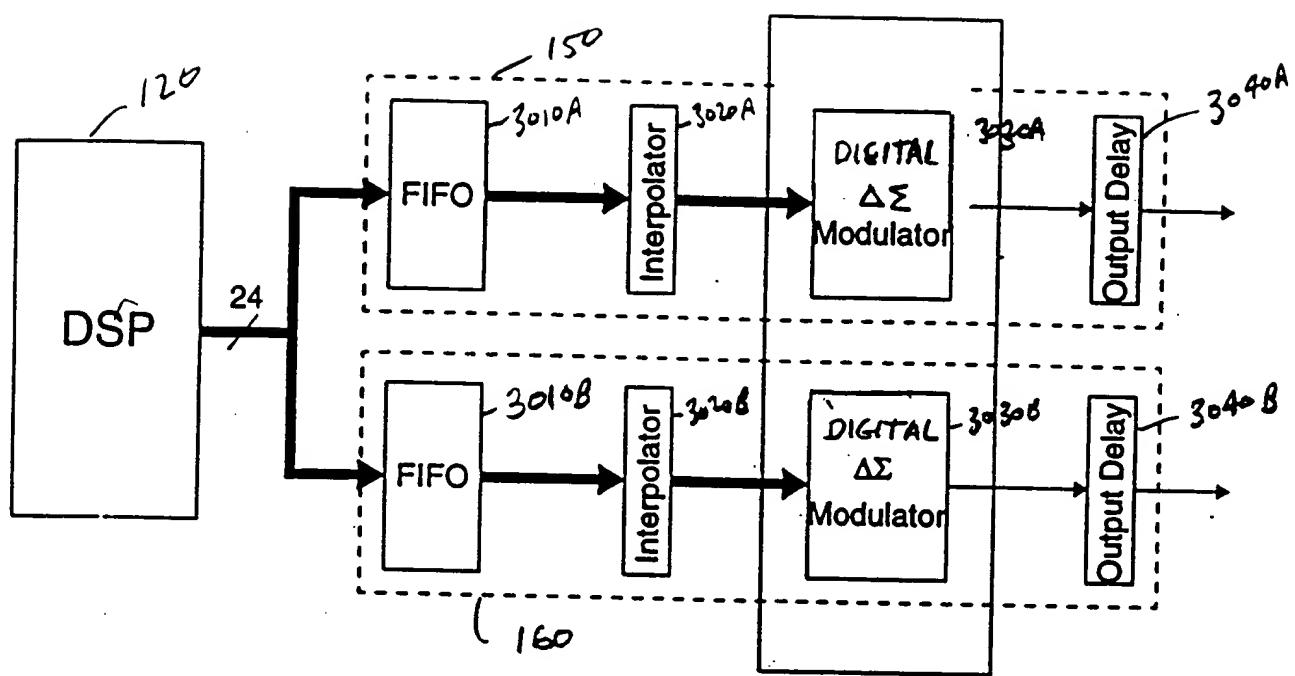
FIGURE 29
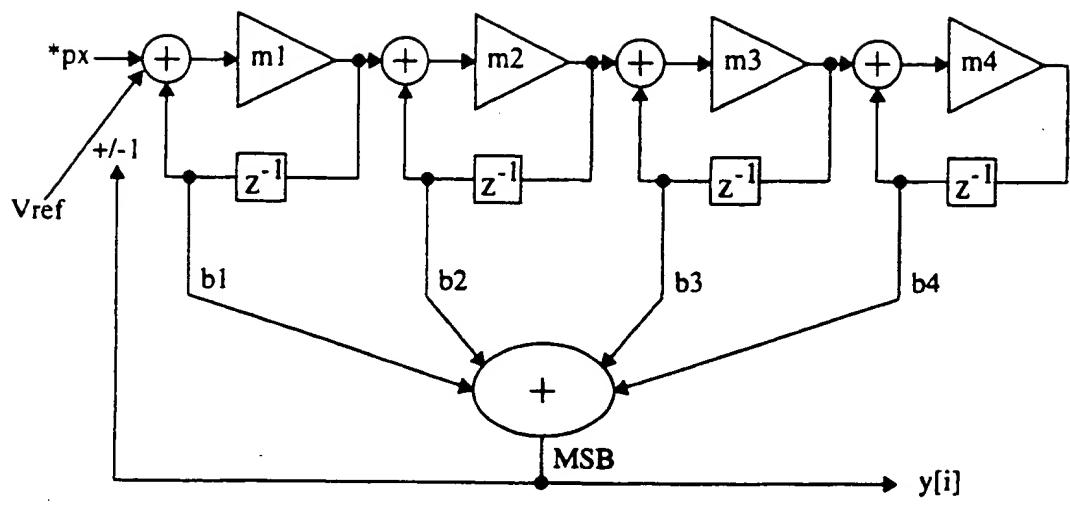
FIGURE 30A

FIGURE 30 B

FIGURE 30C1	———————————	wire
FIGURE 30C2	————⟋————— 24	24 wires
FIGURE 30C3	▭	register
FIGURE 30C4	▽	multiplexer
FIGURE 30C5	▷	tristate buffer
FIGURE 30C6	▷o	inverter
FIGURE 30C7	)▷o	exclusive or gate
FIGURE 30C8	▽+	adder
FIGURE 30C9	▽*	multiplier
FIGURE 30C10	▱	right shifter

D4 D3 D2 D1

+ + + +

B2 B3 B2 B1

* * * *

+ +

DIN

+VREF
-VREF

DIN

+

+

SUM[MSB] → DOUT

FIGURE 30D

D4 D3 D2 D1

+ + + +

>>5

>>7 >>9 >>4 >>5 >>1

+ + + ← 1

+ +

DIN

DIN

+VREF
-VREF

+

+

SUM[MSB] → DOUT
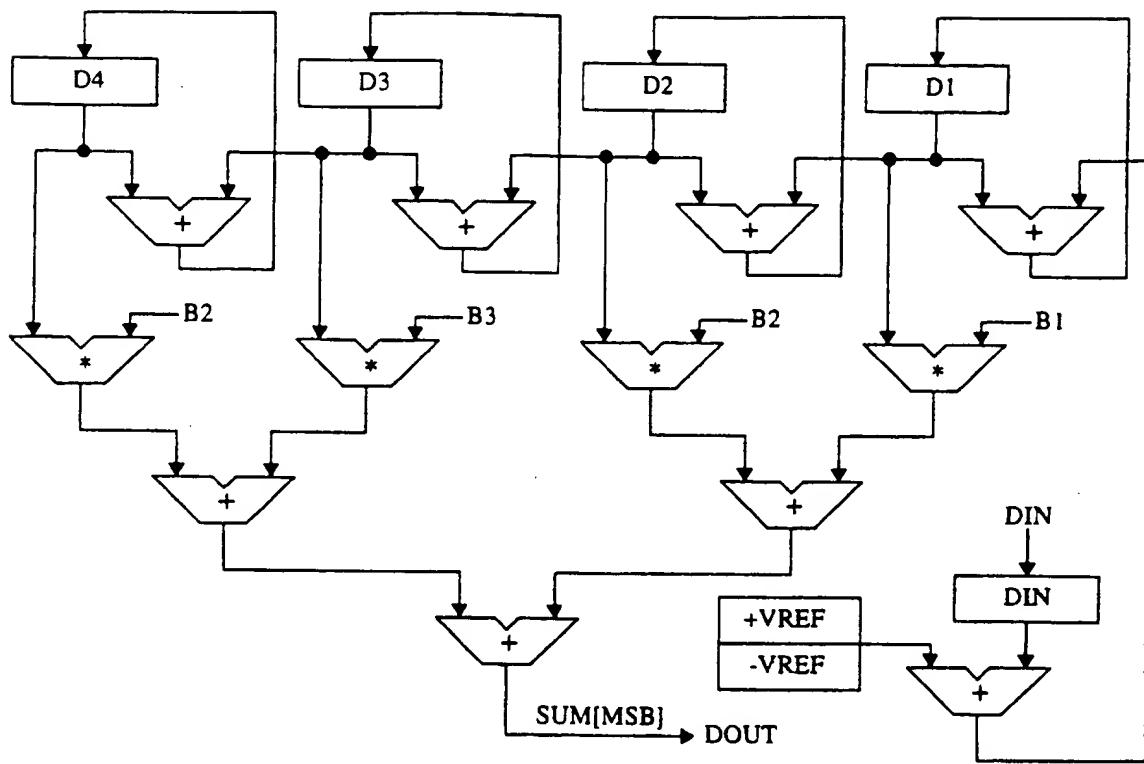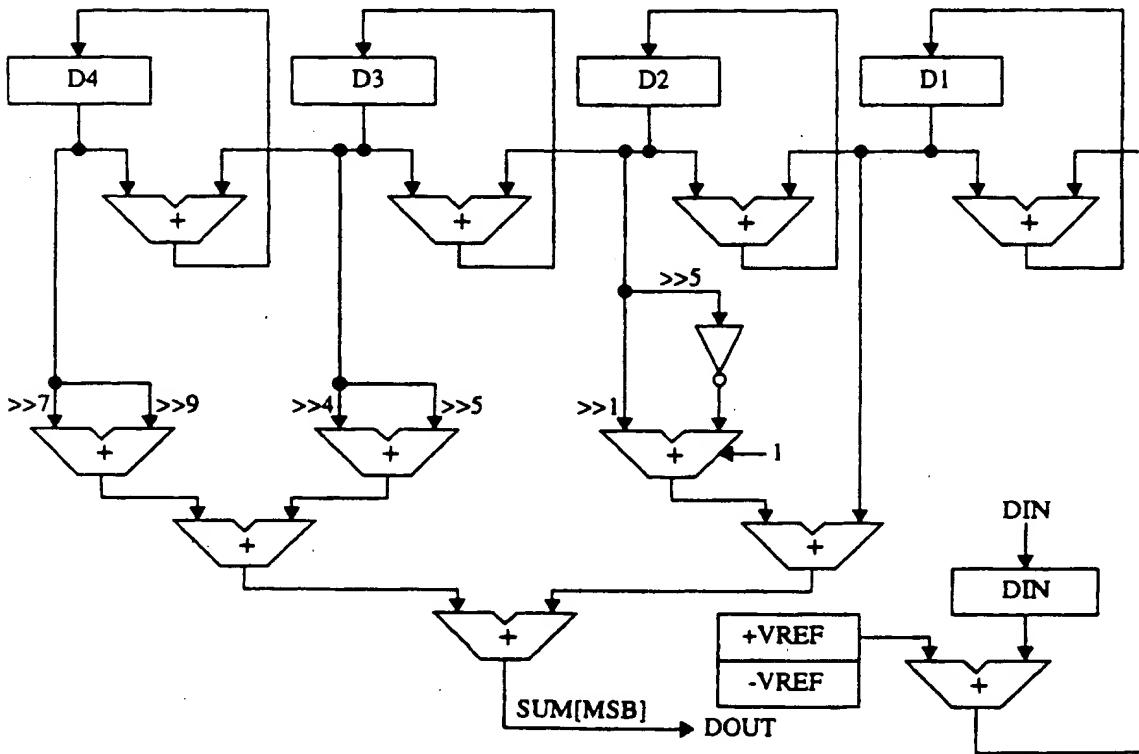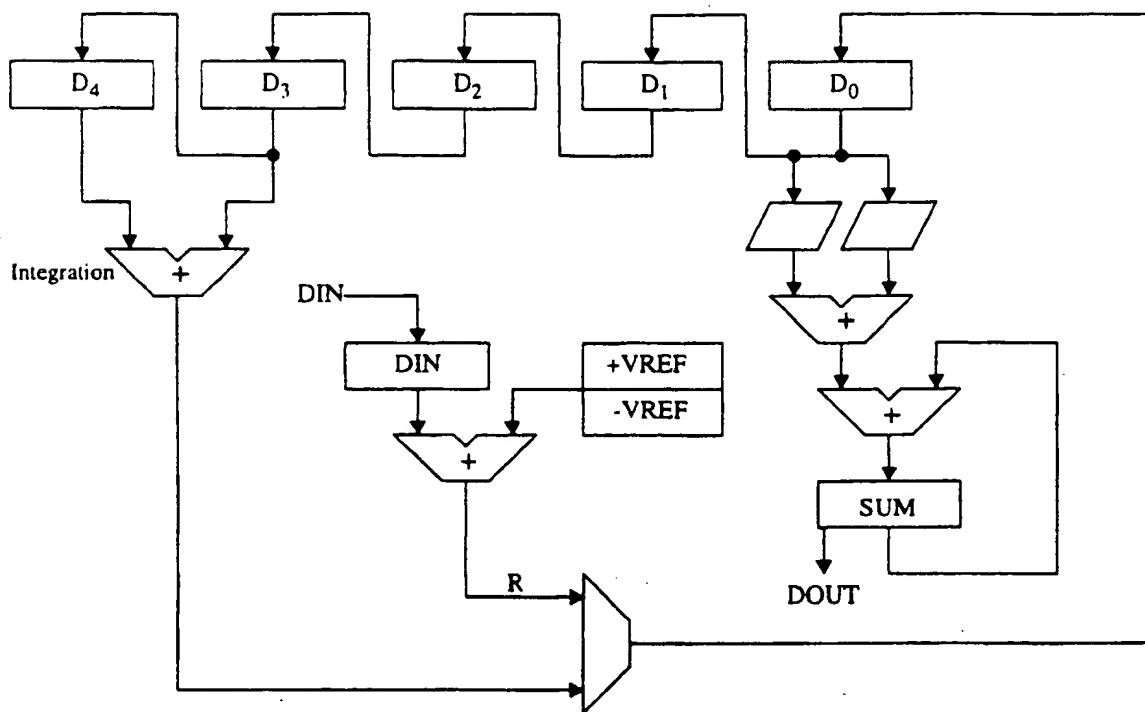
FIGURE 30E

FIGURE 30F1

State	Actions During State		
S0	$D_0(D4_k) = D_4(D4_{k-1}) + D_3(D3_{k-1})$	Clear SUM	Load DIN_k
S1	$D_0(D3_k) = D_4(D3_{k-1}) + D_3(D2_{k-1})$	$SUM_k += D_0(D4_k) >> Shift4$	
S2	$D_0(D2_k) = D_4(D2_{k-1}) + D_3(D1_{k-1})$	$SUM_k += D_0(D3_k) >> Shift3$	
S3	$D_0(D1_k) = D_4(D1_{k-1}) + D_3(R_{k-1})$	$SUM_k += D_0(D2_k) >> Shift2$	
S4		$SUM_k += D_0(D1_k) >> Shift1$	
S5	$D_0(R_k) = DIN_k +/- VREF$		

FIGURE 30F2
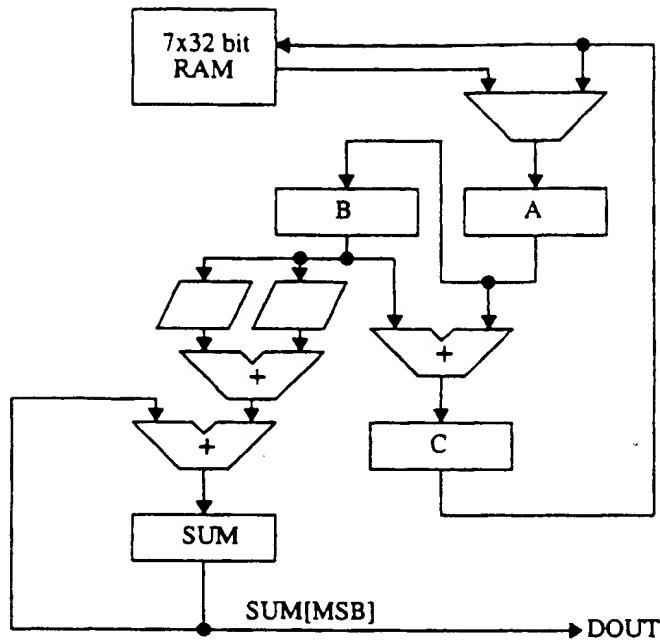
FIGURE 30 G1

State	Actions During State			
S0	Clear SUM	Clear C	Clear B	Clear A
S1				Load A<Mem(D4$_k$)
S2			Shift B<A(D4$_k$)	Load A<Mem(D3$_k$)
S3	SUM$_k$ += B(D4$_k$)>>Shift4	C = B(D4$_k$) + A(D3$_k$)	Shift B<A(D3$_k$)	Load A<Mem(D2$_k$)
S4				Store C>Mem(D4$_{k+1}$)
S5	SUM$_k$ += B(D3$_k$)>>Shift3	C = B(D3$_k$) + A(D2$_k$)	Shift B<A(D2$_k$)	Load A<Mem(D1$_k$)
S6				Store C>Mem(D3$_{k+1}$)
S7	SUM$_k$ += B(D2$_k$)>>Shift2	C = B(D2$_k$) + A(D1$_k$)	Shift B<A(D1$_k$)	Load A<Mem(DIN$_k$)
S8				Store C>Mem(D2$_{k+1}$)
S9	SUM$_k$ += B(D1$_k$)>>Shift1	C = B(D1$_k$) + A(DIN$_k$)	Shift B<A(DIN$_k$)	Load A<Mem(VREF)
S10			Shift B<A(VREF)	LoadReg A<C(Temp)
S11		C = +/-B(VREF) + A(Temp)		
S12				Store C>Mem(D1$_{k+1}$)

FIGURE 30 G2
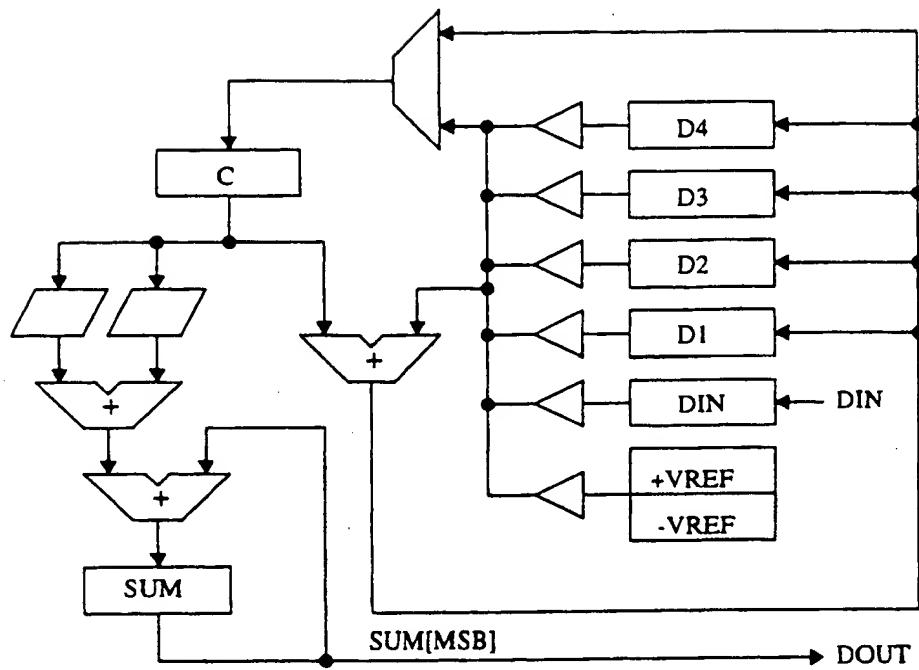
FIGURE 30 H1

| State	Actions During State			
S0	Clear SUM	Load C < $D4_k$		Load DIN_k
S1	SUM_k += $C(D4_k)$>>Shift4	Load C < $D3_k$	$D4_{k+1}$ = $C(D4_k)$ + $D3_k$	
S2	SUM_k += $C(D3_k)$>>Shift3	Load C < $D2_k$	$D3_{k+1}$ = $C(D3_k)$ + $D2_k$	
S3	SUM_k += $C(D2_k)$>>Shift2	Load C < $D1_k$	$D2_{k+1}$ = $C(D2_k)$ + $D1_k$	
S4	SUM_k += $C(D1_k)$>>Shift1	C(Temp) = $C(D1_k)$ + DIN_k		
S5			$D1_{k+1}$ = C(Temp) +/- VREF	

FIGURE 30H2
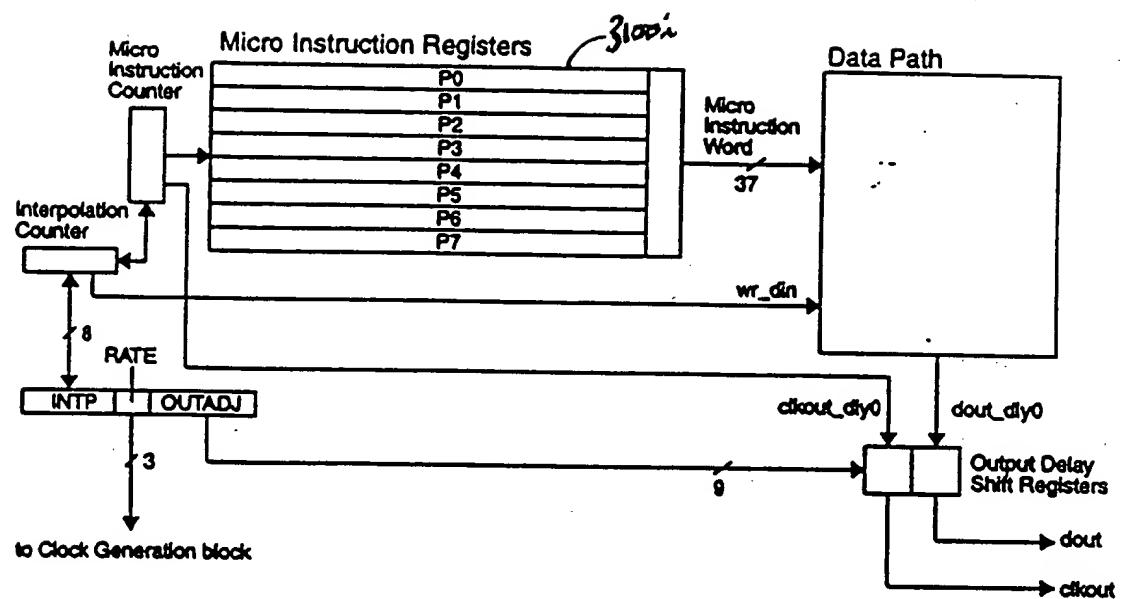
FIGURE 31
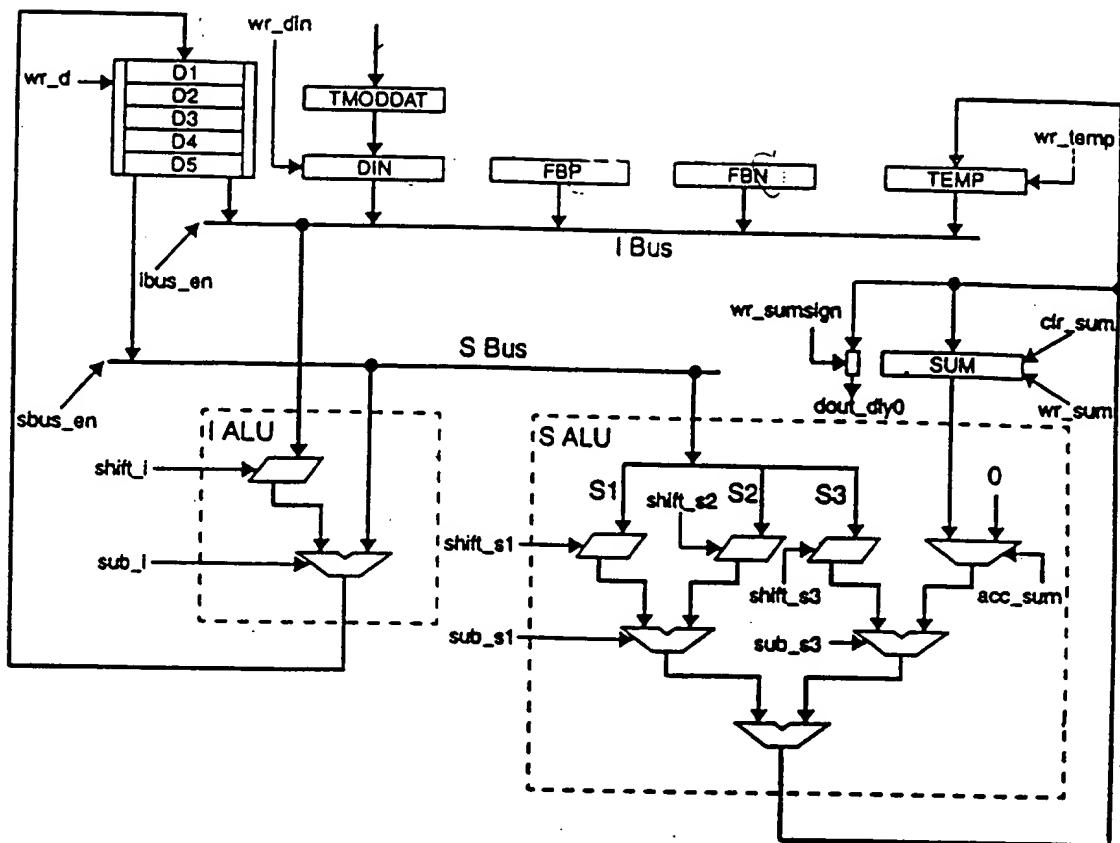
FIGURE 32

P	Feedforward	Integration	Temp	Din	SUM	SUMSIGN	TEMP	S Bus	I Bus	Write I
0	$SUM_k = D4_k \gg 11$ $+ D4_k \gg 9$ $+ D4_k \gg 7$	$D4_{k+1} = D4_k + D3_k$		Load DIN_k	Write			$+D4 \gg 7$ $+D4 \gg 9$ $+D4 \gg 11$	$+D3$	D4
1	$SUM_k = SUM_k$ $+ D3_k \gg 8$ $+ D3_k \gg 5$ $+ D3_k \gg 4$	$D3_{k+1} = D3_k + D2_k$			Acc/ Write			$+D3 \gg 4$ $+D3 \gg 5$ $+D3 \gg 8$	$+D2$	D3
2	$SUM_k = SUM_k$ $+ D2_k \gg 1$ $- D2_k \gg 7$ $- D2_k \gg 4$	$D2_{k+1} = D2_k + D1_k$			Acc/ Write			$-D2 \gg 4$ $+D2 \gg 1$ $-D2 \gg 7$	$+D1$	D2
3	$SUM_k = SUM_k$ $+ D1_k$	$D1_{k+1}' = D1_k + DIN_k$			Acc/ Write	Write		$+D1$ $+D1$ $-D1$	$+DIN$	D1
4		$D1_{k+1} = D1_{k+1}' +/- VREF$							$+FB$	D1
5										
6										
7										

FIGURE 33

Table (rotated in original). Bit positions 0–36 grouped as shown.

HEX	A	s s 3	shift_s3					shift_s2					s s 1	shift_s1					s 1	shift_J					sbus_en			fbus_en			wr_d			T	S	C	G
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36
051C 00E4AC	0	0	1	1	0	1	0	1	0	0	1	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	1	0	1	0	0
04D3 0082A1	1	0	0	0	0	1	0	1	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	1	0	0	1	0	1	1	0	0	1	0	0
048A 00909F	1	1	1	1	1	0	0	1	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	1	0	0	1	0	0
1469 000003	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	1	1	0	0	0	1	0	1
0079 000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	1	1	1	0	0	0	0	0	0
0000 000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0000 000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0000 000000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

FIGURE 34

FIGURE 35A

Table 1: Legend

| dout_dly0	data output bit, 0 delay
dout	data output bit, 0-63 clock delay
delay[5:0]	how many clocks (0-63) to delay output data dout_dly0
delay[5:4]	selects segment into which to direct dout_dly0
delay[3:0]	selects where to tap segment D to get dout

FIGURE 35B

FIGURE 36

RSU & ADC interface Clock Relationships with SYNC

RSU Clocks (Created from CLK16 Rising Edge)

Figure 37